Lógica
Aula 10

Leliane Nunes de Barros

2018

leliane@ime.usp.br

(recordando) Forma Normal Conjuntiva (ou Clausal) (CNF)

Definição de CNF:

- Um literal é um átomo p, ou a negação de um átomo $\neg p$.

(recordando) Forma Normal Conjuntiva (ou Clausal) (CNF)

Definição de CNF:

- Um literal é um átomo p, ou a negação de um átomo $\neg p$.
- A fórmula φ está em forma normal conjuntiva (CNF) se ela for da forma $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$, para algum $n \leq 1$, sendo cada φ_i uma disjunção de literais (cláusula), para todo $i \in \{1, 2, ..., N\}$

(recordando) Forma Normal Conjuntiva (ou Clausal) (CNF)

Definição de CNF:

- Um literal é um átomo p, ou a negação de um átomo $\neg p$.
- A fórmula φ está em forma normal conjuntiva (CNF) se ela for da forma $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$, para algum $n \leq 1$, sendo cada φ_i uma disjunção de literais (cláusula), para todo $i \in \{1, 2, ..., N\}$

Nota: Algumas vezes incluimos o caso $n = 0$, que por convenção representa o termo T

(recordando) Forma Normal Conjuntiva (ou Clausal) (CNF)

Definição de CNF:

- Um literal é um átomo p, ou a negação de um átomo $\neg p$.
- A fórmula φ está em forma normal conjuntiva (CNF) se ela for da forma $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$, para algum $n \leq 1$, sendo cada φ_i uma disjunção de literais (cláusula), para todo $i \in \{1, 2, ..., N\}$

Nota: Algumas vezes incluimos o caso $n = 0$, que por convenção representa o termo T

Exemplos de CNFs:

- $(\neg q \vee p \vee r) \wedge (\neg p \vee r) \wedge q$
- $(p \vee r) \wedge (\neg p \vee r) \wedge (p \vee \neg r)$

(recordando) Forma Normal Conjuntiva (ou Clausal) (CNF)

Definição de CNF:

- Um literal é um átomo p, ou a negação de um átomo $\neg p$.
- A fórmula φ está em forma normal conjuntiva (CNF) se ela for da forma $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$, para algum $n \leq 1$, sendo cada φ_i uma disjunção de literais (cláusula), para todo $i \in \{1, 2, ..., N\}$

Nota: Algumas vezes incluimos o caso $n = 0$, que por convenção representa o termo T

Exemplos de CNFs:

- $(\neg q \vee p \vee r) \wedge (\neg p \vee r) \wedge q$
- $(p \vee r) \wedge (\neg p \vee r) \wedge (p \vee \neg r)$

Exemplo de fórmula que não está em CNF:

- $(\neg(p \vee q) \vee r) \wedge (q \vee r)$

Validade de uma CNF

Fórmulas do tipo CNF produzem provas de validade mais simples. Por exemplo, se quisermos provar a validade de uma CNF $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$:

- basta provar $\varphi_1, \varphi_2, \ldots, \varphi_n$ (pela semântica do $\wedge$).

Validade de uma CNF

Fórmulas do tipo CNF produzem provas de validade mais simples. Por exemplo, se quisermos provar a validade de uma CNF $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$:

- basta provar $\varphi_1, \varphi_2, \ldots, \varphi_n$ (pela semântica do $\wedge$).

- uma disjunção de literais $L_1 \vee L_2 \vee ... L_n$ é válida sse existe i, j com $1 \leq i$ e $j \leq n$ tal que L_i é $\neg L_j$.

Validade de uma CNF

Fórmulas do tipo CNF produzem provas de validade mais simples. Por exemplo, se quisermos provar a validade de uma CNF $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$:

- basta provar $\varphi_1, \varphi_2, \ldots, \varphi_n$ (pela semântica do $\wedge$).

- uma disjunção de literais $L_1 \vee L_2 \vee ...L_n$ é válida sse existe i, j com $1 \leq i$ e $j \leq n$ tal que L_i é $\neg L_j$.

Resumo:

1. Uma fórmula em CNF é válida sse todas as suas cláusulas são válidas.

Validade de uma CNF

Fórmulas do tipo CNF produzem provas de validade mais simples. Por exemplo, se quisermos provar a validade de uma CNF $\varphi_1 \land \varphi_2 \land \ldots \land \varphi_n$:

- basta provar $\varphi_1, \varphi_2, \ldots, \varphi_n$ (pela semântica do $\land$).

- uma disjunção de literais $L_1 \lor L_2 \lor ... L_n$ é válida sse existe i, j com $1 \leq i$ e $j \leq n$ tal que L_i é $\neg L_j$.

Resumo:

1. Uma fórmula em CNF é válida sse todas as suas cláusulas são válidas.
2. Uma cláusula é válida sse ela contém um átomo e sua negação.

Transformar em CNF

Toda fórmula da LP pode ser transformada em CNF!

1. Eliminar $\rightarrow$: $\varphi \rightarrow \psi \equiv \neg\varphi \lor \psi$

Transformar em CNF

Toda fórmula da LP pode ser transformada em CNF!

1. Eliminar $\rightarrow$: $\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$
2. Mover $\neg$ para dentro: $\neg(\varphi \wedge \psi) \equiv \neg\varphi \vee \neg\psi$ e $\neg(\varphi \vee \psi) \equiv \neg\varphi \wedge \neg\psi$

Transformar em CNF

Toda fórmula da LP pode ser transformada em CNF!

1. Eliminar $\rightarrow$: $\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$
2. Mover $\neg$ para dentro: $\neg(\varphi \wedge \psi) \equiv \neg\varphi \vee \neg\psi$ e $\neg(\varphi \vee \psi) \equiv \neg\varphi \wedge \neg\psi$
3. Eliminar dupla negação: $\neg\neg\varphi \equiv \varphi$

Transformar em CNF

Toda fórmula da LP pode ser transformada em CNF!

1. Eliminar $\to$: $\varphi \to \psi \equiv \neg\varphi \vee \psi$
2. Mover $\neg$ para dentro: $\neg(\varphi \wedge \psi) \equiv \neg\varphi \vee \neg\psi$ e $\neg(\varphi \vee \psi) \equiv \neg\varphi \wedge \neg\psi$
3. Eliminar dupla negação: $\neg\neg\varphi \equiv \varphi$
4. Distribuir $\vee$ e $\wedge$: $\varphi \vee (\psi \wedge \chi) \equiv (\varphi \vee \psi) \wedge (\varphi \vee \chi)$

Transformar em CNF

Toda fórmula da LP pode ser transformada em CNF!

1. Eliminar $\rightarrow$: $\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$
2. Mover $\neg$ para dentro: $\neg(\varphi \wedge \psi) \equiv \neg\varphi \vee \neg\psi$ e $\neg(\varphi \vee \psi) \equiv \neg\varphi \wedge \neg\psi$
3. Eliminar dupla negação: $\neg\neg\varphi \equiv \varphi$
4. Distribuir $\vee$ e $\wedge$: $\varphi \vee (\psi \wedge \chi) \equiv (\varphi \vee \psi) \wedge (\varphi \vee \chi)$

Transformar em CNF

Toda fórmula da LP pode ser transformada em CNF!

1. Eliminar $\rightarrow$: $\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$
2. Mover $\neg$ para dentro: $\neg(\varphi \wedge \psi) \equiv \neg\varphi \vee \neg\psi$ e $\neg(\varphi \vee \psi) \equiv \neg\varphi \wedge \neg\psi$
3. Eliminar dupla negação: $\neg\neg\varphi \equiv \varphi$
4. Distribuir $\vee$ e $\wedge$: $\varphi \vee (\psi \wedge \chi) \equiv (\varphi \vee \psi) \wedge (\varphi \vee \chi)$

Exemplo: $(\neg p \wedge q) \rightarrow (p \wedge (r \rightarrow q))$

Por que CNF?

1. Toda fórmula pode ser transformada em CNF.

Por que CNF?

1. Toda fórmula pode ser transformada em CNF.
2. Teorema: $\varphi_1, \ldots, \varphi_n \vdash \psi$ sse $\varphi_1 \wedge \ldots \wedge \varphi_n \wedge \neg\psi$ não é SAT.

O problema SAT

"Dada uma fórmula, decidir se ela é satisfatível."

O problema SAT

"Dada uma fórmula, decidir se ela é satisfatível."

- SAT é NP-completo [Cook 1971]

O problema SAT

"Dada uma fórmula, decidir se ela é satisfatível."

- SAT é NP-completo [Cook 1971]
- Competição desde 2002: `http://www.satcompetition.org/`

O problema SAT

"Dada uma fórmula, decidir se ela é satisfatível."

- SAT é NP-completo [Cook 1971]
- Competição desde 2002: `http://www.satcompetition.org/`

O problema SAT

"Dada uma fórmula, decidir se ela é satisfatível."

- SAT é NP-completo [Cook 1971]
- Competição desde 2002: `http://www.satcompetition.org/`

Valididade, equivalência e derivação podem ser reduzidos a uma problema SAT.

- não existem algoritmos eficientes mas ...

- vários métodos alternativos tentam ser o mais eficiente possível!!

NaiveSAT

Entrada: φ, em CNF

Saída: v, se $v(\varphi) = T$; "não", caso contrário

Para **toda valoração** v sobre os átomos de φ faça:

`se` $v(C) = T$ `então devolva` v

`Devolva "não"`

NaiveSAT

Entrada: φ, em CNF

Saída: v, se $v(\varphi) = T$; "não", caso contrário

> Para **toda valoração** v sobre os átomos de φ faça:
>
> > `se` $v(C) = T$ `então devolva` v
>
> `Devolva "não"`

- Método simples e ineficiente para CNFs grandes
- Trabalha no nível semântico (buscando por valorações)

Método alternativo de reescrita: Resolução

Única regra:

$$\frac{\phi \vee p \qquad \psi \vee \neg p}{\phi \vee \psi}$$

Método alternativo de reescrita: Resolução

Única regra:

$$\frac{\phi \vee p \qquad \psi \vee \neg p}{\phi \vee \psi}$$

- Essa regra generaliza as regras da dedução natural: Modus Ponens ($\rightarrow_e$), Modus Tolens e Eliminação da Negação ($\neg_e$).
- A cláusula vazia pode ser vista como uma disjunção de nenhum disjunto, o que equivale à constante Falso ($\perp$)

Método alternativo de reescrita: Resolução

Única regra:

$$\frac{\phi \lor p \qquad \psi \lor \neg p}{\phi \lor \psi}$$

- Essa regra generaliza as regras da dedução natural: Modus Ponens ($\to_e$), Modus Tolens e Eliminação da Negação ($\neg_e$).
- A cláusula vazia pode ser vista como uma disjunção de nenhum disjunto, o que equivale à constante Falso ($\bot$)

Refutação por resolução (completa)

Raciocínio por contradição:

1. Para provar $\varphi_1, \ldots, \varphi_n \vdash \psi$, transforme $\chi = \varphi_1 \wedge \ldots \wedge \varphi_n \wedge \neg\psi$ em CNF.

Refutação por resolução (completa)

Raciocínio por contradição:

1. Para provar $\varphi_1, \ldots, \varphi_n \vdash \psi$, transforme $\chi = \varphi_1 \wedge \ldots \wedge \varphi_n \wedge \neg\psi$ em CNF.
2. Seja C o conjunto de cláusulas obtido.

Refutação por resolução (completa)

Raciocínio por contradição:

1. Para provar $\varphi_1, \ldots, \varphi_n \vdash \psi$, transforme $\chi = \varphi_1 \wedge \ldots \wedge \varphi_n \wedge \neg\psi$ em CNF.
2. Seja C o conjunto de cláusulas obtido.
3. Aplique resolução ao conjunto quantas vezes for possível.

Refutação por resolução (completa)

Raciocínio por contradição:

1. Para provar $\varphi_1, \ldots, \varphi_n \vdash \psi$, transforme $\chi = \varphi_1 \wedge \ldots \wedge \varphi_n \wedge \neg\psi$ em CNF.
2. Seja C o conjunto de cláusulas obtido.
3. Aplique resolução ao conjunto quantas vezes for possível.
4. Se em algum momento gerar uma cláusula vazia, devolva T (χ não é SAT e portanto $\varphi_1, \ldots, \varphi_n \vdash \psi$ é válido)

Refutação por resolução (completa)

Raciocínio por contradição:

1. Para provar $\varphi_1, \ldots, \varphi_n \vdash \psi$, transforme $\chi = \varphi_1 \wedge \ldots \wedge \varphi_n \wedge \neg\psi$ em CNF.
2. Seja C o conjunto de cláusulas obtido.
3. Aplique resolução ao conjunto quantas vezes for possível.
4. Se em algum momento gerar uma cláusula vazia, devolva T (χ não é SAT e portanto $\varphi_1, \ldots, \varphi_n \vdash \psi$ é válido)
5. Senão, devolva F (χ é SAT).

Refutação por resolução (completa)

Raciocínio por contradição:

1. Para provar $\varphi_1, \ldots, \varphi_n \vdash \psi$, transforme $\chi = \varphi_1 \wedge \ldots \wedge \varphi_n \wedge \neg\psi$ em CNF.
2. Seja C o conjunto de cláusulas obtido.
3. Aplique resolução ao conjunto quantas vezes for possível.
4. Se em algum momento gerar uma cláusula vazia, devolva T (χ não é SAT e portanto $\varphi_1, \ldots, \varphi_n \vdash \psi$ é válido)
5. Senão, devolva F (χ é SAT).

Refutação por resolução (completa)

Raciocínio por contradição:

1. Para provar $\varphi_1, \ldots, \varphi_n \vdash \psi$, transforme $\chi = \varphi_1 \wedge \ldots \wedge \varphi_n \wedge \neg\psi$ em CNF.
2. Seja C o conjunto de cláusulas obtido.
3. Aplique resolução ao conjunto quantas vezes for possível.
4. Se em algum momento gerar uma cláusula vazia, devolva T (χ não é SAT e portanto $\varphi_1, \ldots, \varphi_n \vdash \psi$ é válido)
5. Senão, devolva F (χ é SAT).

Exemplo:

- $p \to q, r \to s \vdash (p \vee r) \to (q \vee s)$

Exemplo passo a passo (transformando em CNF)

$$p \to q, r \to s \vdash (p \vee r) \to (q \vee s)$$

1. Transformar $\chi = (p \to q) \wedge (r \to s) \wedge \neg((p \vee r) \to (q \vee s))$ em CNF:

$(p \to q) \wedge (r \to s) \wedge \neg((p \vee r) \to (q \vee s)) \equiv$

Exemplo passo a passo (transformando em CNF)

$$p \to q, r \to s \vdash (p \lor r) \to (q \lor s)$$

1. Transformar $\chi = (p \to q) \land (r \to s) \land \neg((p \lor r) \to (q \lor s))$ em CNF:

$(p \to q) \land (r \to s) \land \neg((p \lor r) \to (q \lor s)) \equiv$

(eliminando implicações)

$(\neg p \lor q) \land (\neg r \lor s) \land \neg(\neg(p \lor r) \lor (q \lor s)) \equiv$

Exemplo passo a passo (transformando em CNF)

$$p \to q, r \to s \vdash (p \vee r) \to (q \vee s)$$

1. Transformar $\chi = (p \to q) \wedge (r \to s) \wedge \neg((p \vee r) \to (q \vee s))$ em CNF:

$(p \to q) \wedge (r \to s) \wedge \neg((p \vee r) \to (q \vee s)) \equiv$

(eliminando implicações)

$(\neg p \vee q) \wedge (\neg r \vee s) \wedge \neg(\neg(p \vee r) \vee (q \vee s)) \equiv$

(movendo negações para dentro)

$(\neg p \vee q) \wedge (\neg r \vee s) \wedge (\neg\neg(p \vee r) \wedge \neg(q \vee s)) \equiv$

$(\neg p \vee q) \wedge (\neg r \vee s) \wedge \neg\neg(p \vee r) \wedge (\neg q \wedge \neg s) \equiv$

Exemplo passo a passo (transformando em CNF)

$$p \to q, r \to s \vdash (p \vee r) \to (q \vee s)$$

1. Transformar $\chi = (p \to q) \wedge (r \to s) \wedge \neg((p \vee r) \to (q \vee s))$ em CNF:

$(p \to q) \wedge (r \to s) \wedge \neg((p \vee r) \to (q \vee s)) \equiv$

(eliminando implicações)

$(\neg p \vee q) \wedge (\neg r \vee s) \wedge \neg(\neg(p \vee r) \vee (q \vee s)) \equiv$

(movendo negações para dentro)

$(\neg p \vee q) \wedge (\neg r \vee s) \wedge (\neg\neg(p \vee r) \wedge \neg(q \vee s)) \equiv$

$(\neg p \vee q) \wedge (\neg r \vee s) \wedge \neg\neg(p \vee r) \wedge (\neg q \wedge \neg s) \equiv$

(eliminando a dupla negação)

$(\neg p \vee q) \wedge (\neg r \vee s) \wedge (p \vee r) \wedge \neg q \wedge \neg s$

Exemplo passo a passo (resolução)

2. $C = \{\neg p \vee q, \neg r \vee s, p \vee r, \neg q, \neg s\}$

Exemplo passo a passo (resolução)

2. $C = \{\neg p \vee q, \neg r \vee s, p \vee r, \neg q, \neg s\}$

3. Aplicar resolução:

1. $\neg p \vee q$
2. $\neg r \vee s$
3. $p \vee r$
4. $\neg q$
5. $\neg s$
6. $\neg p$ (1,4)
7. r (3,6)
8. $\neg r$ (2,5)
9. [] (7,8)

Exemplo passo a passo

4. Como geramos a cláusula vazia, a fórmula χ não é satisfatível.

Isso quer dizer que o sequente

$$p \to q, r \to s \vdash (p \vee r) \to (q \vee s)$$

É válido.

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

Davis & Putnam, 1960; Davis, Longemann & Loveland, 1962

- Evita construir valorações (modelos) completas eliminado cláusulas e literais sempre que possível.

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

Davis & Putnam, 1960; Davis, Longemann & Loveland, 1962

- Evita construir valorações (modelos) completas eliminado cláusulas e literais sempre que possível.
- *Símbolo puro:* Aparece só positivo ou só negativo. Exemplos:

$$(p \vee \neg r) \wedge (p \vee s) \wedge (t \vee \neg s)$$

$$(\neg r \vee q) \wedge (s \vee p) \wedge (\neg p \vee \neg r) \wedge (n \vee q)$$

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

Davis & Putnam, 1960; Davis, Longemann & Loveland, 1962

- Evita construir valorações (modelos) completas eliminado cláusulas e literais sempre que possível.
- *Símbolo puro:* Aparece só positivo ou só negativo. Exemplos:

$$(p \vee \neg r) \wedge (p \vee s) \wedge (t \vee \neg s)$$

$$(\neg r \vee q) \wedge (s \vee p) \wedge (\neg p \vee \neg r) \wedge (n \vee q)$$

- *Propagação Unitária:* Preferência por cláusulas com um só literal.

$$(\neg p \vee r) \wedge (\neg s) \wedge (t \vee s \vee p)$$

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

Davis & Putnam, 1960; Davis, Longemann & Loveland, 1962

- Evita construir valorações (modelos) completas eliminado cláusulas e literais sempre que possível.
- *Símbolo puro:* Aparece só positivo ou só negativo. Exemplos:

$$(p \vee \neg r) \wedge (p \vee s) \wedge (t \vee \neg s)$$

$$(\neg r \vee q) \wedge (s \vee p) \wedge (\neg p \vee \neg r) \wedge (n \vee q)$$

- *Propagação Unitária:* Preferência por cláusulas com um só literal.

$$(\neg p \vee r) \wedge (\neg s) \wedge (t \vee s \vee p)$$

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

Davis & Putnam, 1960; Davis, Longemann & Loveland, 1962

- Evita construir valorações (modelos) completas eliminado cláusulas e literais sempre que possível.
- *Símbolo puro:* Aparece só positivo ou só negativo. Exemplos:

$$(p \vee \neg r) \wedge (p \vee s) \wedge (t \vee \neg s)$$

$$(\neg r \vee q) \wedge (s \vee p) \wedge (\neg p \vee \neg r) \wedge (n \vee q)$$

- *Propagação Unitária:* Preferência por cláusulas com um só literal.

$$(\neg p \vee r) \wedge (\neg s) \wedge (t \vee s \vee p) \equiv (\neg p \vee r) \wedge (t \vee p)$$

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

1. Começa com modelo vazio.

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

1. Começa com modelo vazio.
2. Se alguma cláusula é F, devolve F.

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

1. Começa com modelo vazio.
2. Se alguma cláusula é F, devolve F.
3. Se todas as cláusulas são T, devolve T.

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

1. Começa com modelo vazio.
2. Se alguma cláusula é F, devolve F.
3. Se todas as cláusulas são T, devolve T.
4. Remove todas as cláusulas com símbolo puro.

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

1. Começa com modelo vazio.
2. Se alguma cláusula é F, devolve F.
3. Se todas as cláusulas são T, devolve T.
4. Remove todas as cláusulas com símbolo puro.
5. Quando tem cláusulas unitárias, acrescenta ao modelo.

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

1. Começa com modelo vazio.
2. Se alguma cláusula é F, devolve F.
3. Se todas as cláusulas são T, devolve T.
4. Remove todas as cláusulas com símbolo puro.
5. Quando tem cláusulas unitárias, acrescenta ao modelo.
6. Se não, **escolhe** literal l e acrescenta ao modelo. (ponto de retrocesso)

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

1. Começa com modelo vazio.
2. Se alguma cláusula é F, devolve F.
3. Se todas as cláusulas são T, devolve T.
4. Remove todas as cláusulas com símbolo puro.
5. Quando tem cláusulas unitárias, acrescenta ao modelo.
6. Se não, **escolhe** literal l e acrescenta ao modelo. (ponto de retrocesso)
7. Apaga as cláusulas contendo l.

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

1. Começa com modelo vazio.
2. Se alguma cláusula é F, devolve F.
3. Se todas as cláusulas são T, devolve T.
4. Remove todas as cláusulas com símbolo puro.
5. Quando tem cláusulas unitárias, acrescenta ao modelo.
6. Se não, **escolhe** literal l e acrescenta ao modelo. (ponto de retrocesso)
7. Apaga as cláusulas contendo l.
8. Apaga o oposto de l das outras cláusulas.

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

1. Começa com modelo vazio.
2. Se alguma cláusula é F, devolve F.
3. Se todas as cláusulas são T, devolve T.
4. Remove todas as cláusulas com símbolo puro.
5. Quando tem cláusulas unitárias, acrescenta ao modelo.
6. Se não, **escolhe** literal l e acrescenta ao modelo. (ponto de retrocesso)
7. Apaga as cláusulas contendo l.
8. Apaga o oposto de l das outras cláusulas.
9. Repete até achar contradição ou não ter mais escolhas (conjunto de cláusulas vazio).

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

1. Começa com modelo vazio.
2. Se alguma cláusula é F, devolve F.
3. Se todas as cláusulas são T, devolve T.
4. Remove todas as cláusulas com símbolo puro.
5. Quando tem cláusulas unitárias, acrescenta ao modelo.
6. Se não, **escolhe** literal l e acrescenta ao modelo. (ponto de retrocesso)
7. Apaga as cláusulas contendo l.
8. Apaga o oposto de l das outras cláusulas.
9. Repete até achar contradição ou não ter mais escolhas (conjunto de cláusulas vazio).
10. Se o modelo tem contradição, a fórmula não é T. Se não, é uma valoração.

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

1. Começa com modelo vazio.
2. Se alguma cláusula é F, devolve F.
3. Se todas as cláusulas são T, devolve T.
4. Remove todas as cláusulas com símbolo puro.
5. Quando tem cláusulas unitárias, acrescenta ao modelo.
6. Se não, **escolhe** literal l e acrescenta ao modelo. (ponto de retrocesso)
7. Apaga as cláusulas contendo l.
8. Apaga o oposto de l das outras cláusulas.
9. Repete até achar contradição ou não ter mais escolhas (conjunto de cláusulas vazio).
10. Se o modelo tem contradição, a fórmula não é T. Se não, é uma valoração.

O Algoritmo DPLL (Davis–Putnam–Logemann–Loveland)

1. Começa com modelo vazio.
2. Se alguma cláusula é F, devolve F.
3. Se todas as cláusulas são T, devolve T.
4. Remove todas as cláusulas com símbolo puro.
5. Quando tem cláusulas unitárias, acrescenta ao modelo.
6. Se não, **escolhe** literal l e acrescenta ao modelo. (ponto de retrocesso)
7. Apaga as cláusulas contendo l.
8. Apaga o oposto de l das outras cláusulas.
9. Repete até achar contradição ou não ter mais escolhas (conjunto de cláusulas vazio).
10. Se o modelo tem contradição, a fórmula não é T. Se não, é uma valoração.